

NAME: NIMISH KUSHWAHA

REG. NO.: CH.EN.U4CSE22073

IMPLEMENTATION OF A QUEUE USING ARRAYS IN C LANGUAGE

1. PROGRAM CODE:

```
1  #include<stdio.h>
2  #define MAX_SIZE 100
3
4  int queue[MAX_SIZE];
5  int front = -1;
6  int rear = -1;
7
8  int isEmpty(){
9      return (front == -1 && rear == -1);
10 }
11 int isFull(){
12     return (rear == MAX_SIZE -1);
13 }
14 void enqueue(int value){
15     if (isFull()){
16         printf("Queue is Full. Enqueue opeation is failed.\n");
17         return;
18     }
19
20     else if(isEmpty()){
21         front = 0;
22         rear = 0;
23     }
24     else{
25         rear++;
26     }
27
28     queue[rear] = value;
29     printf("%d enqueued sucessfully.\n", value);
30 }
```

```

31 // Function to dequeue an element from the front of the queue
32 int dequeue(){
33     int value;
34     if (isEmpty()){
35         printf("Queue is empty. Dequeue operation failed.\n");
36         return -1;
37     }
38     else {
39         value = queue[front];
40         if(front == rear){
41             front = -1;
42             rear = -1;
43         } else {
44             front++;
45         }
46         return value;
47     }
48 }
49 //function to display the elements in the queue
50 void display(){
51     if(isEmpty()){
52         printf("Queue is empty.\n");
53     }else{
54         printf("Queue elements: ");
55         for(int i = front; i<=rear; i++){
56             printf("%d ", queue[i]);
57         }
58         printf("\n");
59     }
60 }
61 int main(){
62     enqueue(10);
63     enqueue(20);
64     enqueue(30);
65     display(); //Queue elements: 10 20 30
66
67     int dequeuedValue = dequeue();
68     printf("Dequeued value: %d\n", dequeuedValue); //Dequeued value: 10
69     return 0;
70 }

```

2. OUTPUT:

```
PS D:\My Code\FDS> gcc q.c
PS D:\My Code\FDS> ./a.exe
10 enqueued sucessfully.
20 enqueued sucessfully.
30 enqueued sucessfully.
Queue elements: 10 20 30
Dequeued value: 10
PS D:\My Code\FDS> █
```

IMPLEMENTATION OF A QUEUE USING LINKED LIST IN C LANGUAGE

1. PROGRAM CODE:

```
1  #include<stdio.h>
2  #include<stdlib.h>
3
4  //Node structure for the linked list
5  struct Node{
6      int data;
7      struct Node* next;
8  };
9
10 struct Node* front = NULL; //Front pointer of the queue
11 struct Node* rear = NULL; //Rear pointer of the queue
12
13 //Function to check if the queue is empty
14 int isEmpty(){
15     return (front == NULL);
16 }
17
18 //Function to enqueue an element to the rear of the queue
19 void enqueue(int value){
20     struct Node* newNode = (struct Node*)malloc(sizeof(struct Node));
21     newNode->data = value;
22     newNode->next = NULL;
23
24     if (rear == NULL){
25         front = newNode;
26         rear = newNode;
27     } else{
28         rear->next = newNode;
29         rear = newNode;
30     }
31     printf("\n%d enqueued successfully.", value);
32 }
```

```

34 //Function to dequeue an element from the front of the queue
35 int dequeue(){
36     if(isEmpty()){
37         printf("Queue is empty. Dequeue operation failed.\n");
38         return -1;
39     } else{
40         struct Node* temp = front;
41         int value = front->data;
42         front = front->next;
43         free(temp);
44
45         if(front == NULL){
46             rear = NULL;
47         }
48         return value;
49     }
50 }
51
52 //Function to display the elements in the queue
53 void display(){
54     if (isEmpty()){
55         printf("Queue is empty.\n");
56     } else {
57         printf("\nQueue elements: ");
58         struct Node* temp = front;
59         while (temp != NULL){
60             printf("%d ", temp->data);
61             temp = temp->next;
62         }
63         printf("\n");
64     }
65 }
66
67 int main(){
68     enqueue(10);
69     enqueue(20);
70     enqueue(30);
71     display(); //Queue elements: 10 20 30
72
73     int dequeuedValue = dequeue();
74     printf("Dequeued value: %d\n", dequeuedValue); //Dequeued value: 10
75     printf("\n");
76
77     return 0;
78 }

```

2. OUTPUT:

```
PS D:\My Code\FDS> gcc queue.c
```

```
PS D:\My Code\FDS> ./a.exe
```

```
10 enqueued successfully.
```

```
20 enqueued successfully.
```

```
30 enqueued successfully.
```

```
Queue elements: 10 20 30
```

```
Dequeued value: 10
```

IMPLEMENTATION OF A STACK USING AN ARRAY IN C LANGUAGE:

1. PROGRAM CODE:

```
1  #include<stdio.h>
2  int stack[100],choice,n,top,x,i;
3  void push(void);
4  void pop(void);
5  void display(void);
6
7  ..
7  void push()
8  {
9      if(top>=n-1)
10     {
11         printf("\nSTACK is over flow");
12     }
13     else
14     {
15         printf(" Enter a value to be pushed:");
16         scanf("%d",&x);
17         top++;
18         stack[top]=x;
19     }
20 }
21
22
23 void pop()
24 {
25     if(top<=-1)
26     {
27         printf("\n Stack is under flow");
28     }
29     else
30     {
31         printf("\n The popped elements is %d",stack[top]);
32         top--;
33     }
34 }
```

```

36 void display()
37 {
38     if(top>=0)
39     {
40         printf("\n The elements in STACK \n");
41         for(i=top; i>=0; i--)
42             printf("\n%d",stack[i]);
43         printf("\n Press Next Choice");
44     }
45     else
46     {
47         printf("\n The STACK is empty");
48     }
49 }
50
51
52 int main()
53 {
54     top=-1;
55     printf("\n Enter the size of STACK[MAX=100]:");
56     scanf("%d",&n);
57     printf("\nSTACK OPERATIONS USING ARRAY");
58     printf("\n 1.PUSH\n 2.POP\n 3.DISPLAY\n 4.EXIT");
59     do
60     {
61         printf("\nEnter the Choice:");
62         scanf("%d",&choice);
63         switch(choice)
64         {
65             case 1:
66             {
67                 push();
68                 break;
69             }
70             case 2:
71             {
72                 pop();
73                 break;
74             }
75             case 3:
76             {
77                 display();
78                 break;
79             }
80             case 4:
81             {
82                 printf("\nEXIT POINT ");
83                 break;
84             }
85             default:
86             {
87                 printf ("\n Please Enter a Valid Choice(1/2/3/4)");
88             }
89         }
90     }
91     while(choice!=4);
92     return 0;
93 }
94
95

```


2. OUTPUT:

```
Enter the size of STACK[MAX=100]:4
```

```
STACK OPERATIONS USING ARRAY
```

- 1.PUSH
- 2.POP
- 3.DISPLAY
- 4.EXIT

```
Enter the Choice:1
```

```
Enter a value to be pushed:5
```

```
Enter the Choice:1
```

```
Enter a value to be pushed:4
```

```
Enter the Choice:1
```

```
Enter a value to be pushed:3
```

```
Enter the Choice:1
```

```
Enter a value to be pushed:2
```

```
Enter the Choice:3
```

```
The elements in STACK
```

```
2  
3  
4  
5
```

```
Press Next Choice
```

```
Enter the Choice:2
```

```
The popped elements is 2
```

```
Enter the Choice:3
```

```
The elements in STACK
```

```
3  
4  
5
```

```
Press Next Choice
```

```
Enter the Choice:2
```

```
The popped elements is 3
```

```
Enter the Choice:3
```

```
The elements in STACK
```

```
4  
5
```

```
Press Next Choice
```

```
Enter the Choice:4
```

```
EXIT POINT
```

```
PS D:\Study Materials\2nd Sem\19CSE111 (FDS)\Exercise in C> █
```